Online machine learning with River 🌊

Max Halford — 2022 GAIA conference

Online machine learning

Going big

Either improve the hardware

- GPUs
- Clusters

Either improve the software

- Databases - Snowflake, DuckDB
- Analytics - Arrow, Vaex
- Machine learning - PyTorch, RAPIDS

Going online

Handle data as it arrives

Computation becomes stateful

Past data doesn’t have to be revisited

Examples

- Databases - Kafka, RedPanda
- Analytics - Materialize, ksqlDB, Flink
- Machine learning - Vowpal Wabbit, River

Materialize

kafka

Flink

VOWPAL WABBIT

Streaming is the frontier…

… especially for machine learning

But it's not a replacement for batch

- It **might** make sense for certain use cases
- It **might** help you scale
- It **might** make what you're doing simpler

I also think it's elegant, but that's a detail

Online machine learning

An ML system does two things

- Inference
- Learning

Online ML is about doing this online

- One sample at a time
- Limited memory
- No assumptions about the data

🔮 Online inference

Batch models can do it

Easy to scale

A lot of available software

Model selection is challenging

💪 Online learning

Most models can’t learn online

Online models learn one by one

No need to revisit past data

No assumptions about the data

✨ Many benefits

Low memory footprint

Close to reality

Robust to concept drift

Real-time monitoring

🤨 And yet, online < batch

Online models can’t do vectorization

Online models have less capacity

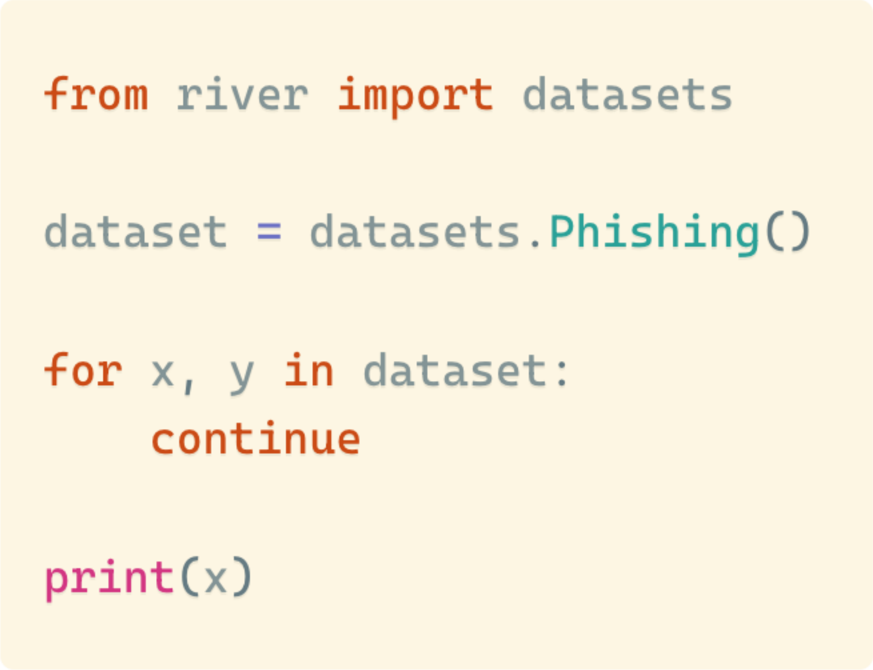
More batch libraries available

Online learning less well known

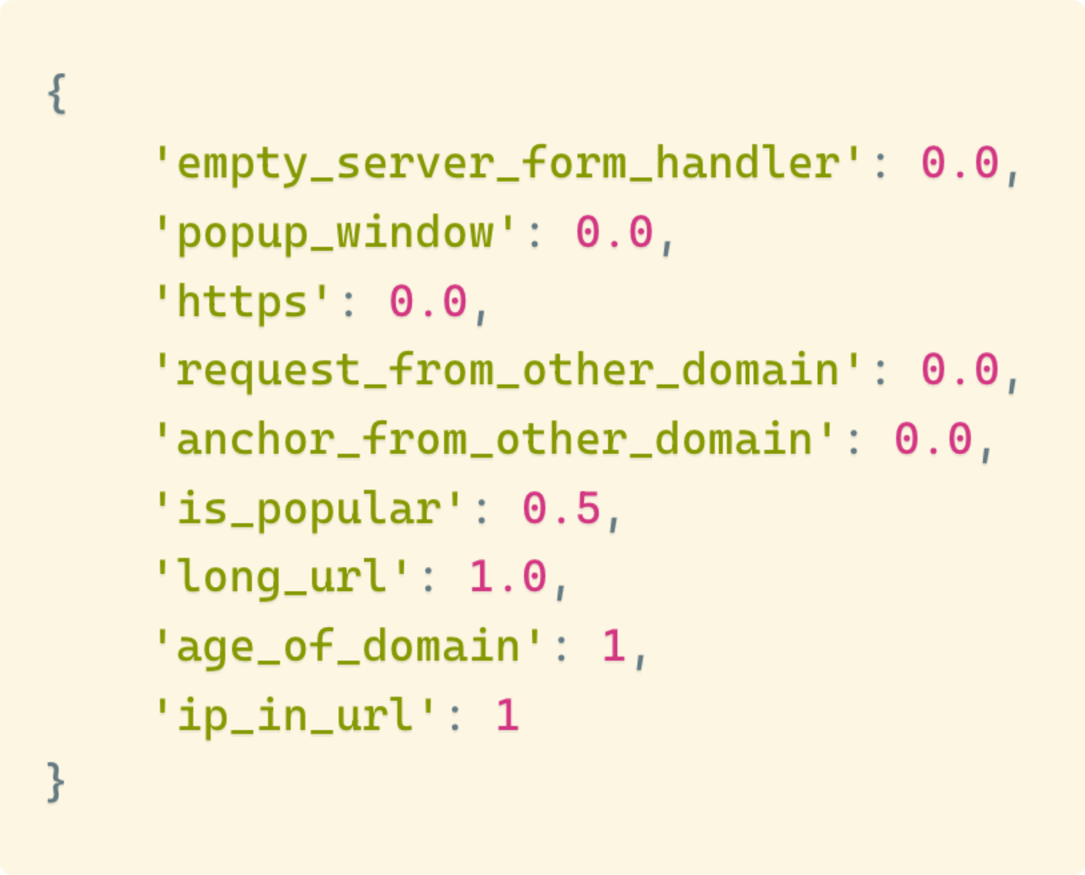
Lack of convincing online examples

I decided to do something about it!

River



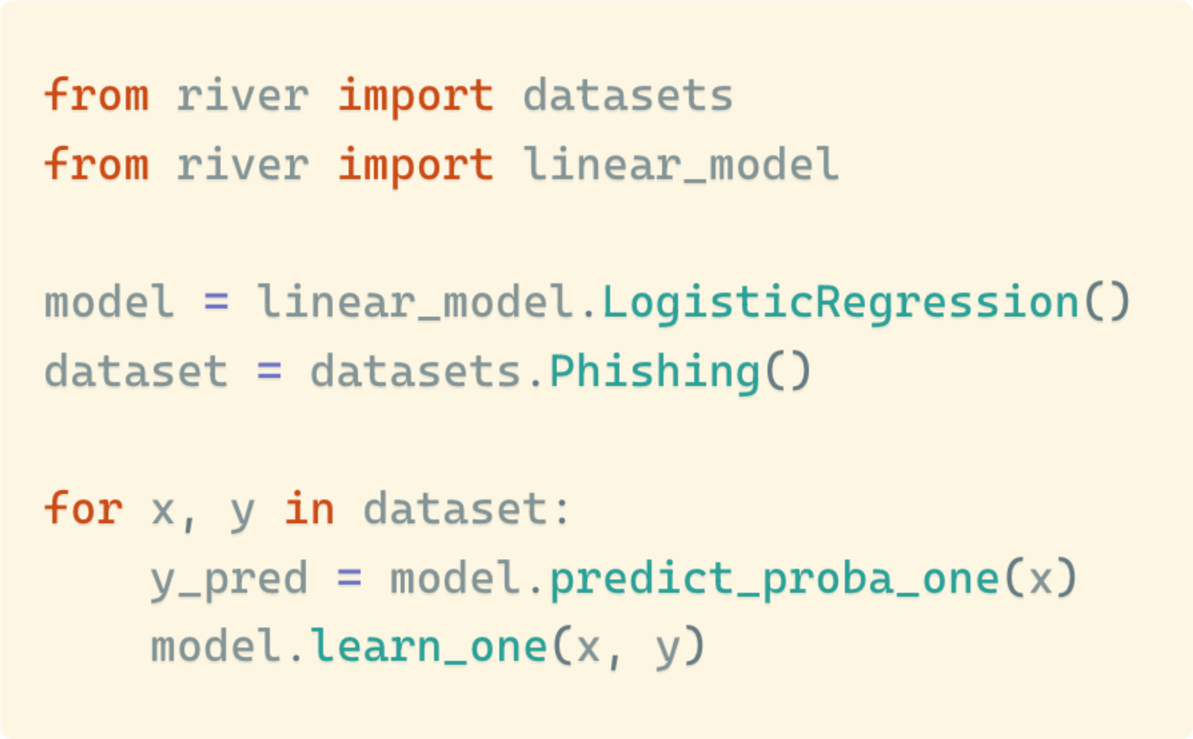
```python
from river import datasets

dataset = datasets.Phishing()

for x, y in dataset:
    continue

print(x)
```



```
{
    'empty_server_form_handler': 0.0,
    'popup_window': 0.0,
    'https': 0.0,
    'request_from_other_domain': 0.0,
    'anchor_from_other_domain': 0.0,
    'is_popular': 0.5,
    'long_url': 1.0,
    'age_of_domain': 1,
    'ip_in_url': 1
}
```

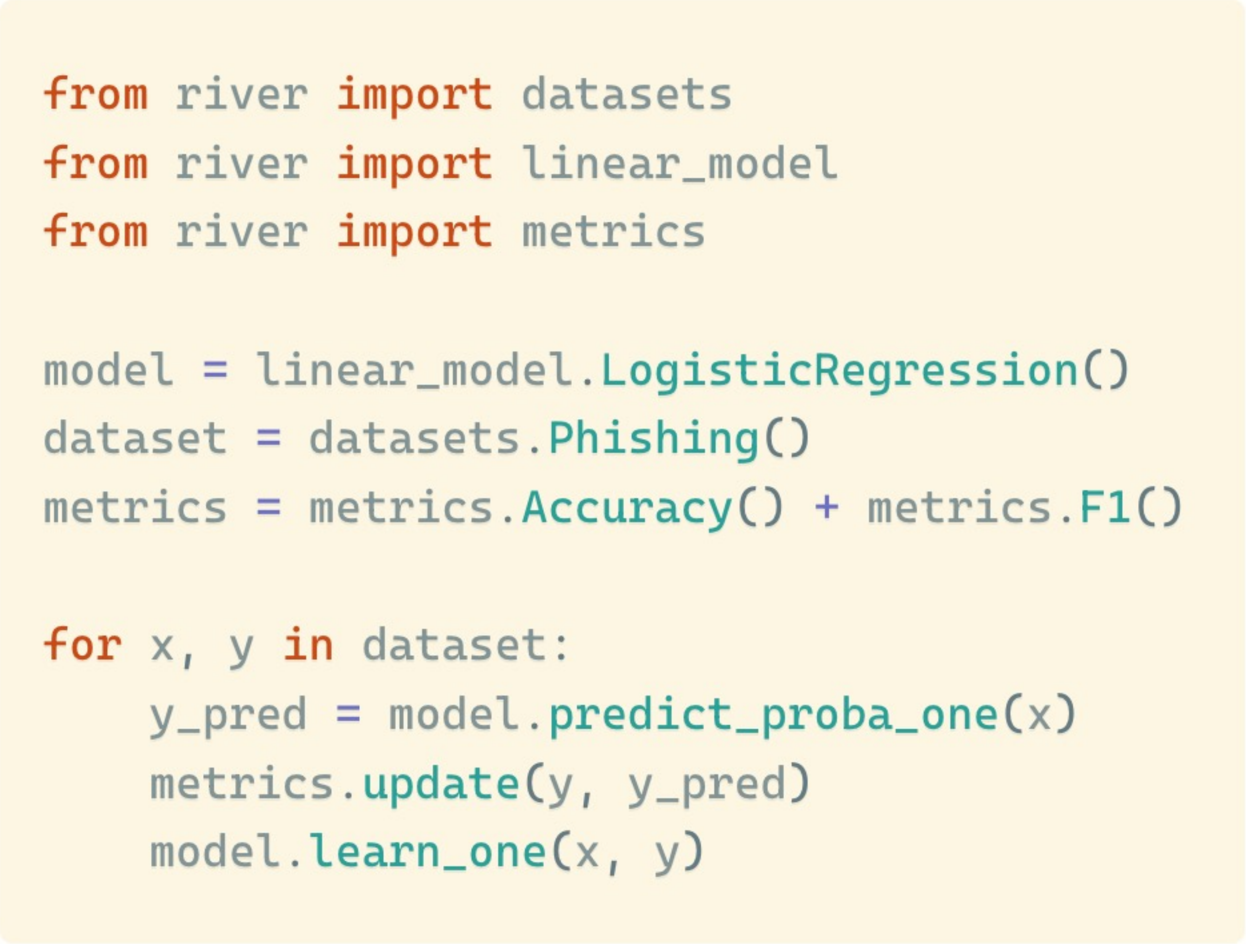
Datasets

Training loop



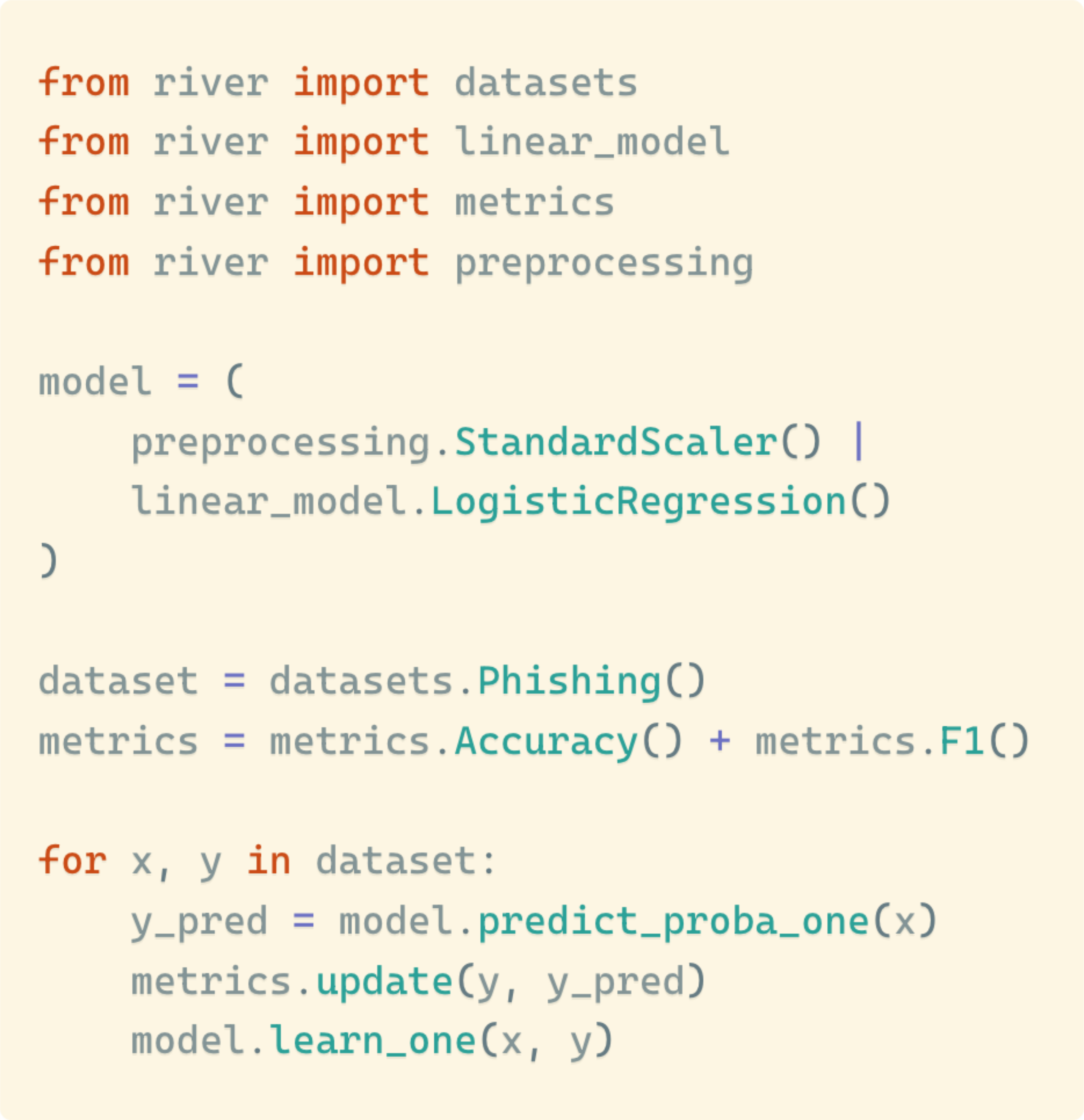
```python
from river import datasets
from river import linear_model

model = linear_model.LogisticRegression()
dataset = datasets.Phishing()

for x, y in dataset:
    y_pred = model.predict_proba_one(x)
    model.learn_one(x, y)
```

Measuring performance



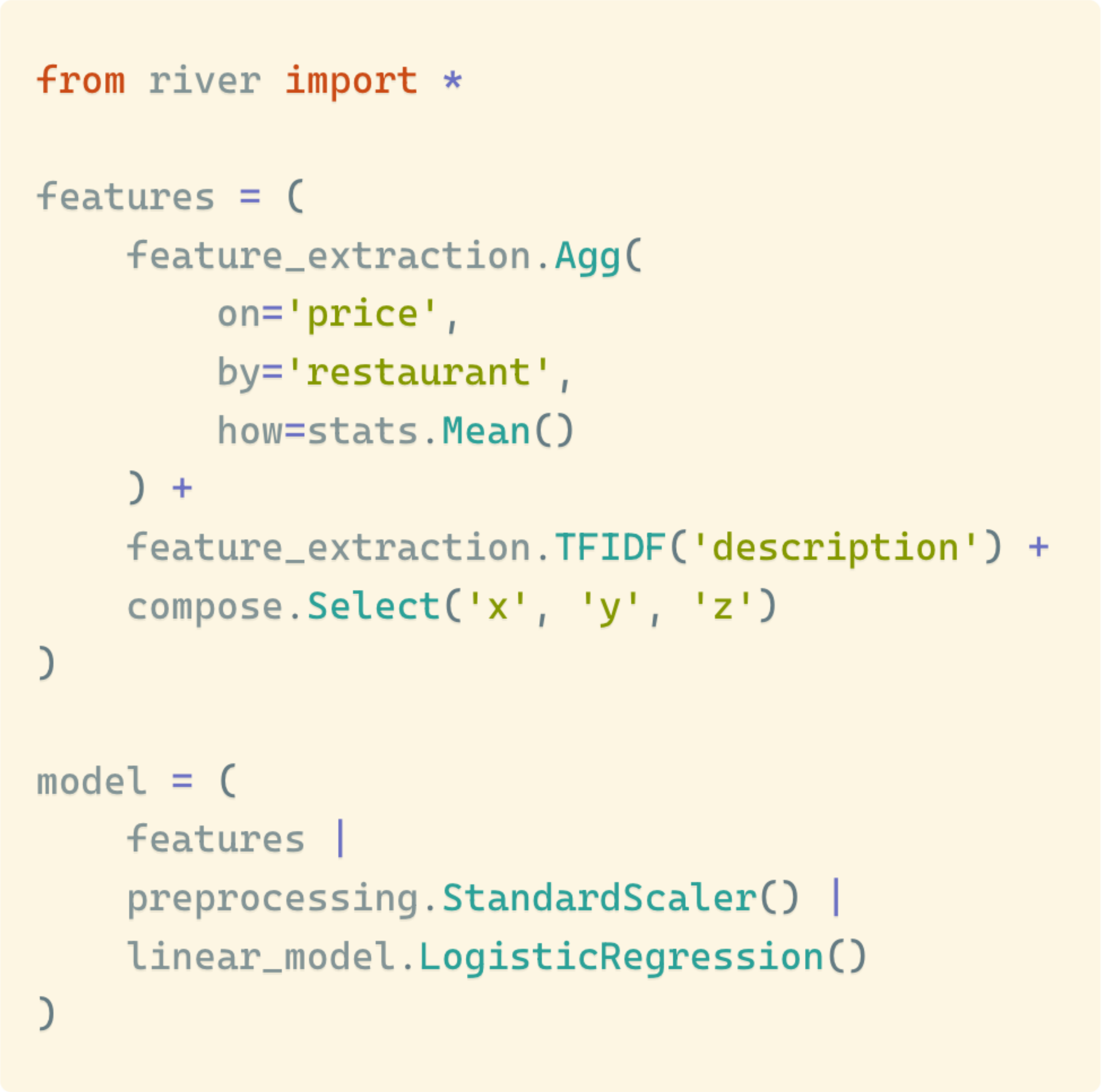
```python
from river import datasets
from river import linear_model
from river import metrics

model = linear_model.LogisticRegression()
dataset = datasets.Phishing()
metrics = metrics.Accuracy() + metrics.F1()

for x, y in dataset:
    y_pred = model.predict_proba_one(x)
    metrics.update(y, y_pred)
    model.learn_one(x, y)
```

Model composition



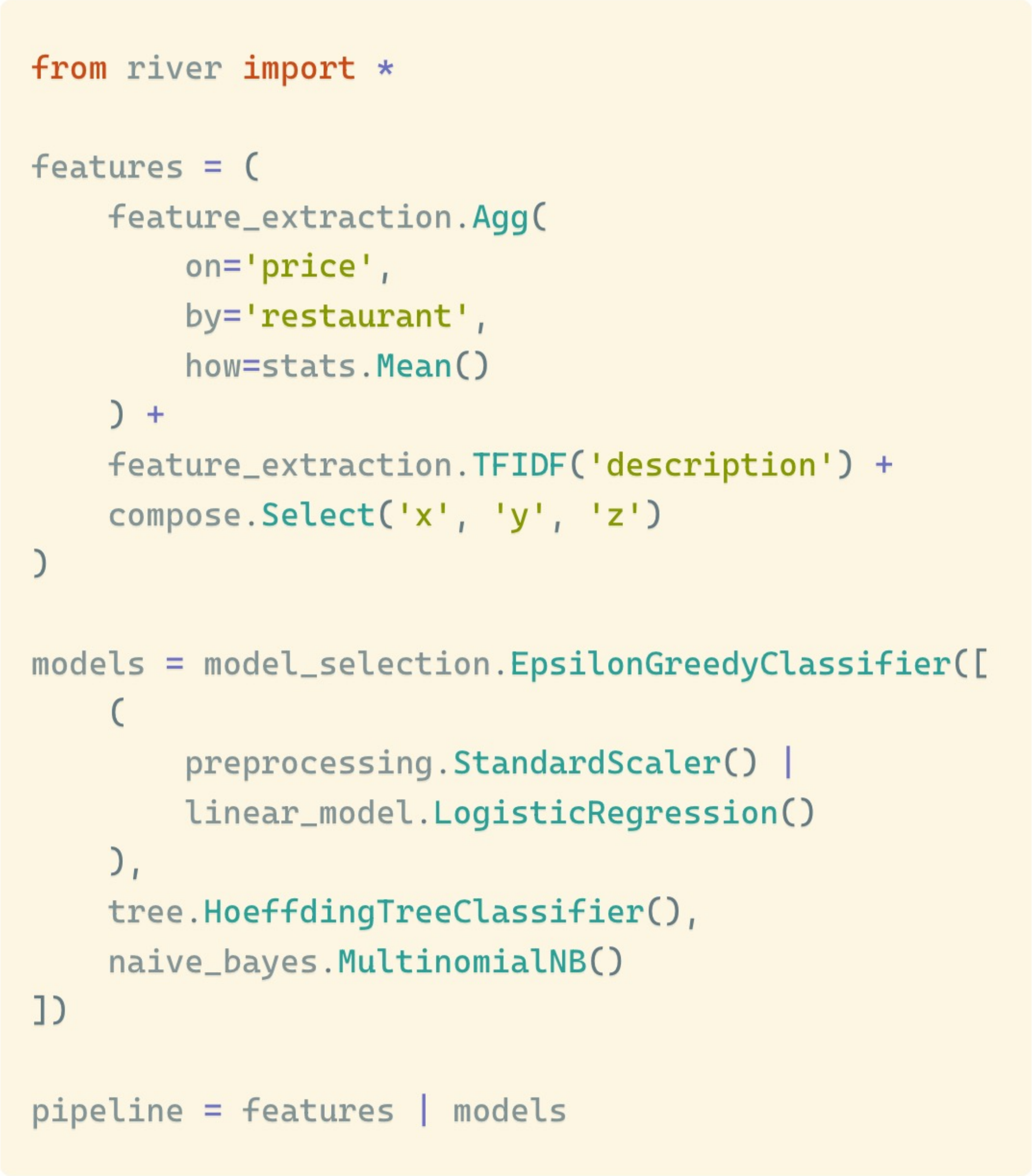
```
from river import datasets
from river import linear_model
from river import metrics
from river import preprocessing

model = (
    preprocessing.StandardScaler() |
    linear_model.LogisticRegression()
)

dataset = datasets.Phishing()
metrics = metrics.Accuracy() + metrics.F1()

for x, y in dataset:
    y_pred = model.predict_proba_one(x)
    metrics.update(y, y_pred)
    model.learn_one(x, y)
```

Feature extraction



```
from river import *

features = (
    feature_extraction.Agg(
        on='price',
        by='restaurant',
        how=stats.Mean()
    ) +
    feature_extraction.TFIDF('description') +
    compose.Select('x', 'y', 'z')
)

model = (
    features |
    preprocessing.StandardScaler() |
    linear_model.LogisticRegression()
)
```

Model selection



```python
from river import *

features = (
    feature_extraction.Agg(
        on='price',
        by='restaurant',
        how=stats.Mean()
    ) +
    feature_extraction.TFIDF('description') +
    compose.Select('x', 'y', 'z')
)

models = model_selection.EpsilonGreedyClassifier([
    (
        preprocessing.StandardScaler() |
        linear_model.LogisticRegression()
    ),
    tree.HoeffdingTreeClassifier(),
    naive_bayes.MultinomialNB()
])

pipeline = features | models
```

Some figures

25k
lines of code

2.5k
unit tests

130
estimators

3
core developers

20
months old

Many modules for many use cases

Bandits

Ranking

Preprocessing

Online statistics

Feature extraction

Anomaly detection

Multi-output learning

Random forests

Naive Bayes

Clustering

Time series forecasting

Online metrics

Linear models

Nearest neighbors

Imbalanced learning

Model selection

Decision trees

FACTORIZATION MACHINES

Neural networks

Production matters

Some companies use River

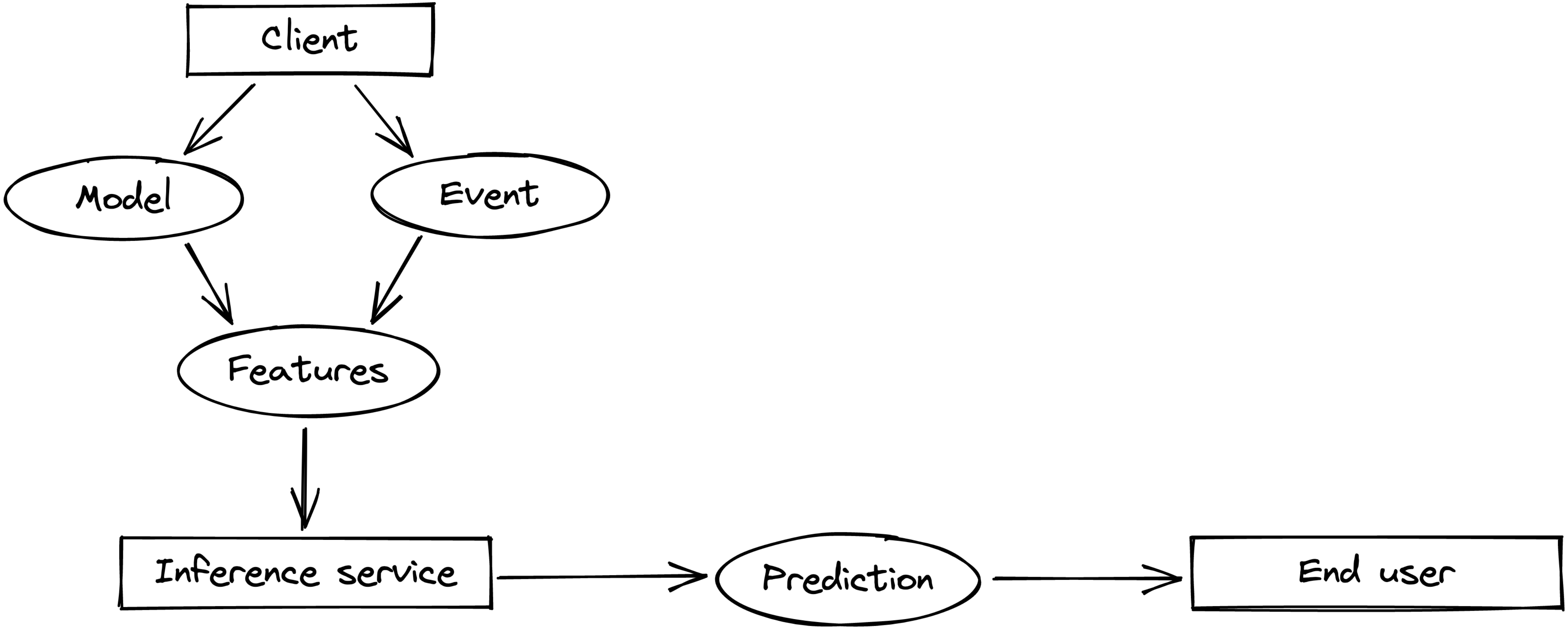
No canonical way to deploy

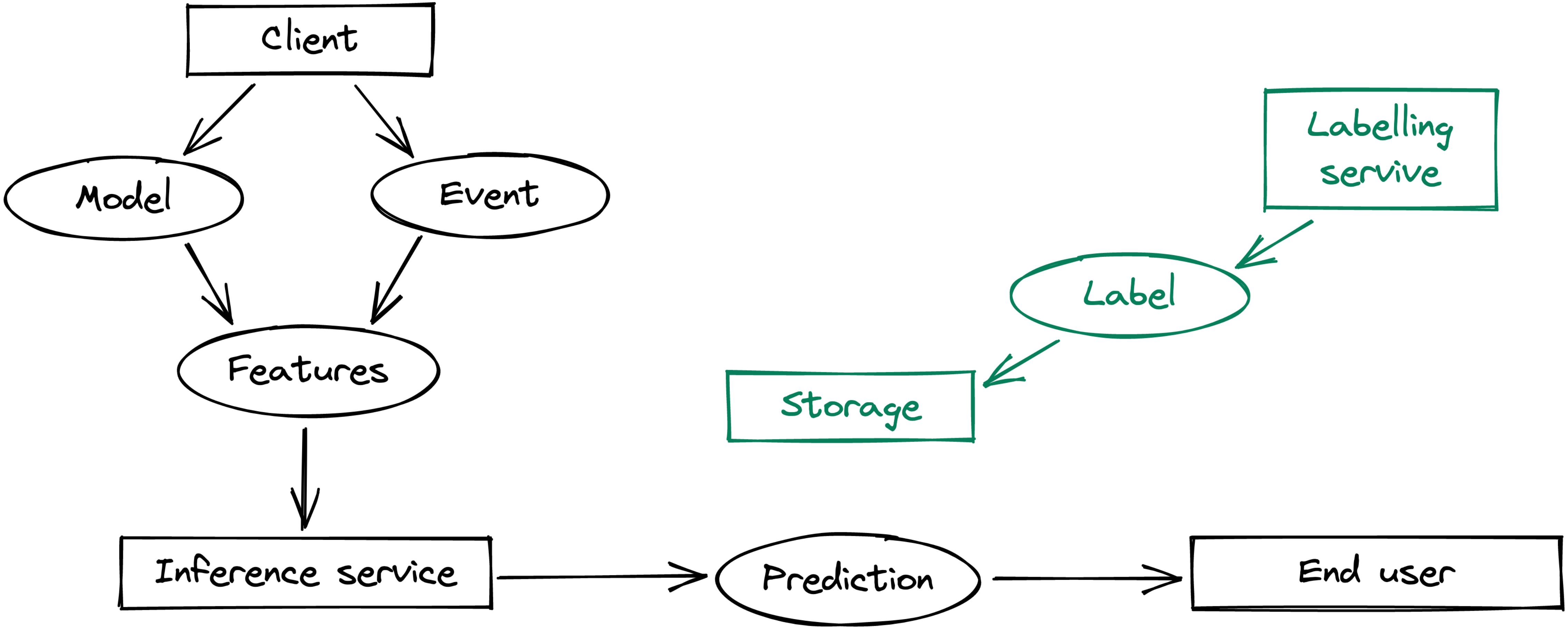
River is not an MLOps tool

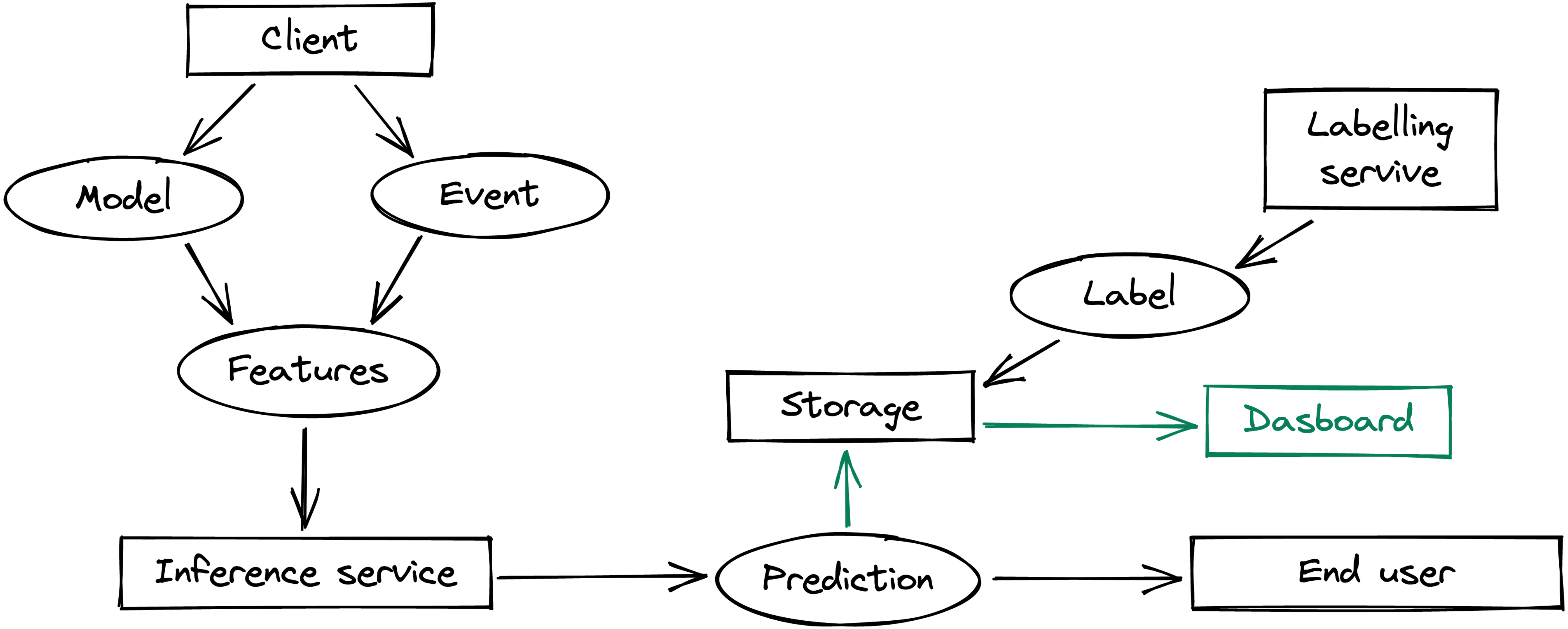
Gap in the ecosystem

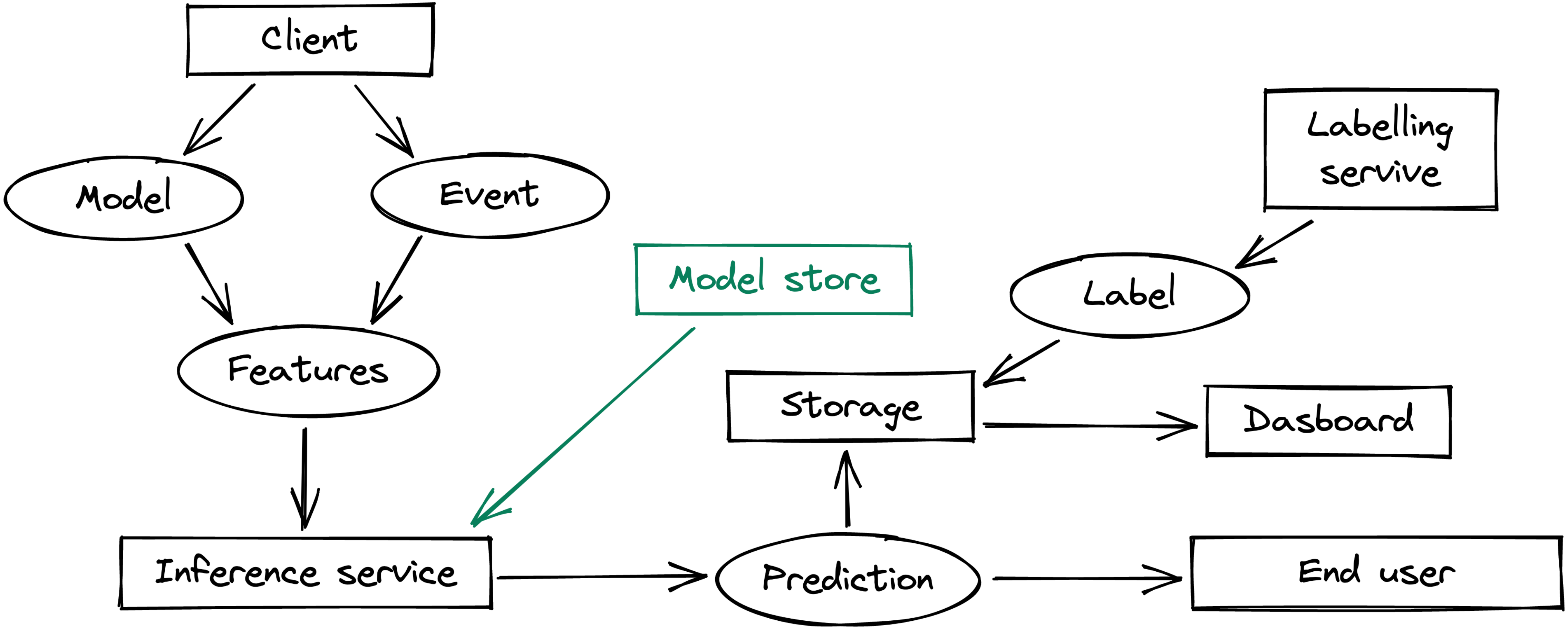
I decided to do something about it! (again)

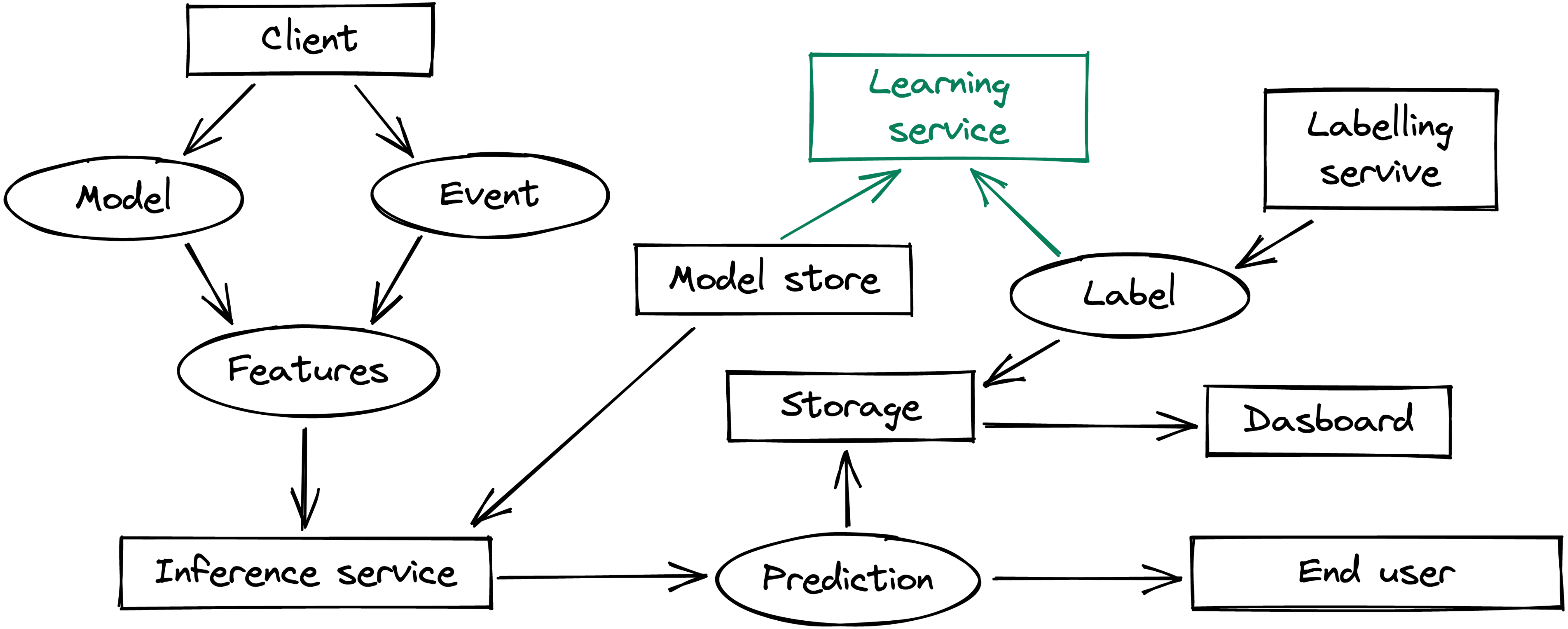
MLOps


Client
Model
Event
Features
Inference service
Prediction
End user


Client
Model
Event
Features
Inference service
Prediction
End user
Labelling
servive
Label
Storage


Client
Model
Event
Labelling servive
Label
Features
Storage
Dasboard
Inference service
Prediction
End user


Client
Model
Event
Features
Model store
Labelling
servive
Label
Storage
Dasboard
Inference service
Prediction
End user


Client
Model
Event
Features
Inference service
Learning service
Model store
Labelling servive
Label
Storage
Dasboard
Prediction
End user

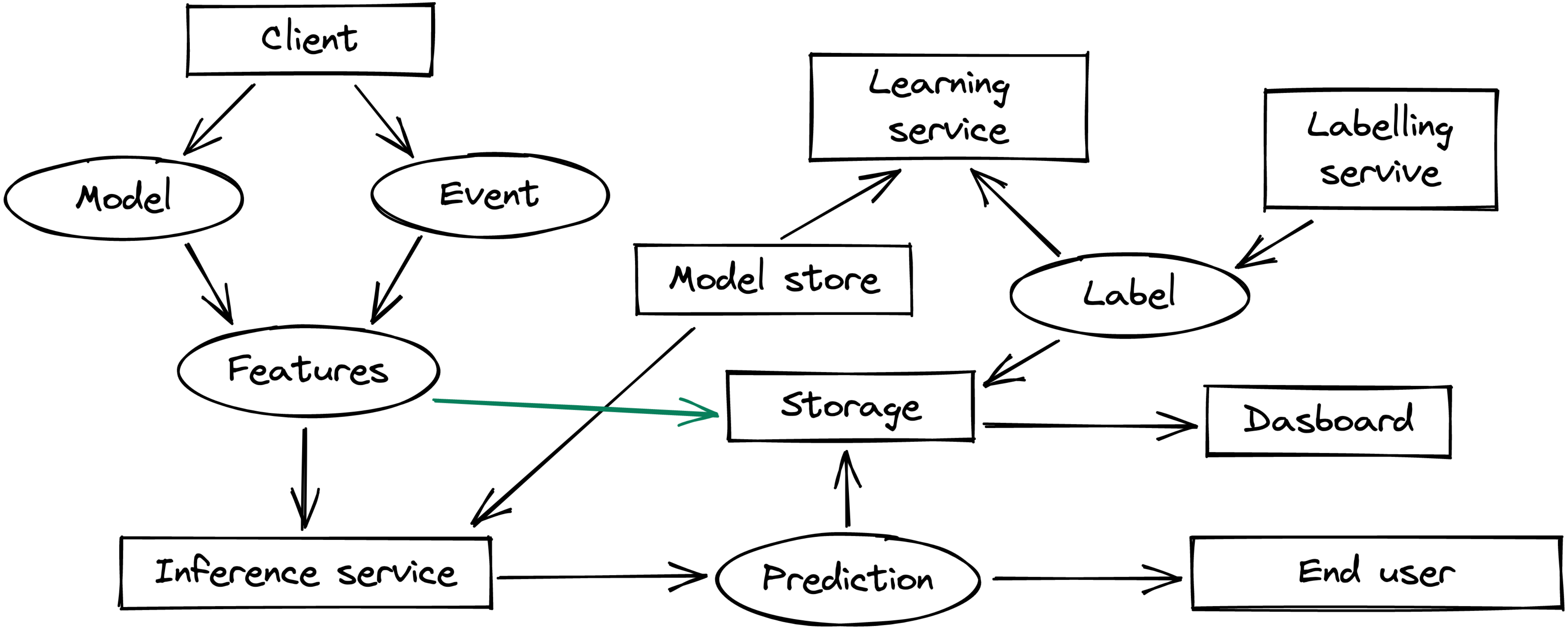
Log and wait strategy

Use same features everywhere

Requires feature storage

Features are joined with labels

Prevents data leakage


Client
Model
Event
Features
Inference service
Learning service
Labelling servive
Model store
Label
Storage
Dasboard
Prediction
End user

Next steps

Still in blueprint phase

Idea is to be technology agnostic

github.com/online-ml/beaver

Feel welcome to reach out 🤗

Max Halford

- 🌱 Data scientist @ Carbonfact
- 🎓 PhD in applied ML
- 🏆 Kaggle competitions master
- 🤓 Online ML became a hobby
- 🧭 maxhalford.github.io

Special thanks to Leonard Aukea,
Geoffrey Bolmier and
Josef Lindman Hörnlund